IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of

CRUTCHER

Atty. Ref.: 2452-13

Serial No. 09/484,455

TC/A.U.: 2141

Filed: 18 January 2000

Examiner: Coulter, Kenneth R.

For: APPLET EMBEDDED CROSS-PLATFORM CACHING

* * * * * * * * * * * *

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Declaration Under 37 C.F.R. 1.131

I, Craig D. Crutcher, declare as follows:

1. I am the inventor of the subject matter claimed in the patent application identified above.

2. I understand that the Patent Office has rejected certain claims of this patent application as allegedly being "anticipated" under 35 USC 102(e) by U.S. Patent No. 6,536,035 to Hawkins filed on 9/3/99.

3. I conceived of the subject matter disclosed and claimed in my patent application before Hawkins' 9/3/99 effective filing date.

4. The computer listing excerpts attached as Exhibit A corroborate my earlier conception of my invention. These listings bear initial dates of before 9/3/99 (the actual dates have been redacted to protect confidentiality).

CRUTCHER
Serial No. 09/484,455

6. Attached Exhibit A further demonstrates that I exercised due diligence from before 9/3/99 through a subsequent reduction to practice of my invention in the form of a "beta" version.

7. I declare further that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Date: 4/6/2004 Craig D. Crutcher

REDACTED

EXHIBIT A

JPM_FILELOG.TXT

//Users/craige/ev30/com/wrq/jpm/CacheEntry.java
... #4 change 4861 edit on [redacted] 22:01:47 by craige@cc_ev3_loader (text) 'Changed Context refs in Context'
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/CLoader.java
... #6 change 5071 edit on [redacted] 16:13:22 by craige@cc_ev3_loader (text) 'I think it sorta works! '
... #5 change 4916 edit on [redacted] 15:58:40 by craige@cc_ev3_loader (text) 'Removed a bit of unused code. '
... #4 change 4701 edit on [redacted] 16:25:28 by craige@cc_ev3_loader (text) 'yet more changes '
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/DefaultSecure.java
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/IClassLoader.java
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/ISecure.java
... #1 change 4687 add on [redacted] 9:40:09 by craige@cc_ev3_loader (text) 'ISecure added. '
//Users/craige/ev30/com/wrq/jpm/JarCache.java
... #3 change 4672 edit on [redacted] 5:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/JarLoader.java
... #4 change 4964 edit on [redacted] 11:18:42 by craige@cc_ev3_loader (text) 'Slight d/l optimization. '
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/jpmresource/Handler.java
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/jpmresource/JPMResourceConnection.java
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/MSCLoader.java
... #5 change 4916 edit on [redacted] 15:58:40 by craige@cc_ev3_loader (text) 'Removed a bit of unused code. '
... #4 change 4701 edit on [redacted] 16:25:28 by craige@cc_ev3_loader (text) 'yet more changes '
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/MSSecure.java
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/NSSecure.java
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '
//Users/craige/ev30/com/wrq/jpm/Secure.java
... #4 change 4917 edit on [redacted] 15:59:09 by doyle@doyle-ev3-loader (text) ''
... #3 change 4672 edit on [redacted] 15:17:01 by craige@cc_ev3_loader (text) 'Yet more changes! '
... #2 change 4617 edit on [redacted] 09:51:26 by craige@cc_ev3_loader (text) 'loader tweaking '
... #1 change 4496 add on [redacted] 14:43:10 by craige@cc_ev3_loader (text) 'JAR package manager classes. '

REDACTED

JarLoader Class Revision History

1 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 5547 [redacted]
craigc@cc_ev3_merge add (text) Here's the first incarnation of the JarLoader. May it rest in peace.
2 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 5586 [redacted]
craigc@cc_ev3_trunk edit (text) JarLoader.FLG_PERSIST changed to JarLoader.FLG_DONT_PERSIST. (So that persistence becomes the default)
3 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 5941 [redacted]
craigc@cc_ev3_trunk edit (text) Added compression and signing to jar loader.
4 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 5943 [redacted]
craigc@cc_ev3_trunk edit (text) Oops.
5 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 5948 [redacted]
craigc@cc_ev3_trunk edit (text) increased IO buffers
6 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 5954 [redacted]
craigc@cc_ev3_trunk edit (text) oops.
7 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 6081 [redacted]
craigc@cc_ev3_trunk edit (text) Tweaks
8 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 6110 [redacted]
craigc@cc_ev3_trunk edit (text) Localized the exception strings
9 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 6573 [redacted]
craigc@cc_ev3_trunk edit (text) renamed public key file, plus added support for a development key.
10 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 6594 [redacted]
craigc@cc_ev3_trunk edit (text) 116160 - now using devJAW.pbk and JAW.pbk and development and production keys respectively
11 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 6733 [redacted]
craigc@cc_ev3_trunk edit (text) Added debug timing statements...REMOVE BEFORE ICO!
12 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 7127 [redacted]
craigc@cc_ev3_trunk edit (text) JarLoader touch up.
13 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 7481 [redacted] doyle@doyle-trunk
edit (text) Core cosntant pruning
14 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 7514 [redacted]
craigc@cc_ev3_trunk edit (text) 116031 - moved jar cache from user.home/wrq_cache -> user.home/wrq/cache
15 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 7594 [redacted] garyh@GARYH4
edit (text+ko) Get the JavaScript API to finally start working.
16 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 7654 [redacted]
craigc@cc_ev3_trunk edit (text+ko) 116827 - a work in progress...should get us into beta, though.
17 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 7691 [redacted]
craigc@cc_ev3_trunk edit (text+ko) debug message
18 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 7737 [redacted]
craigc@cc_ev3_trunk edit (text+ko) 116905 - well, learn something new every day. like, the fact that mkdirs() will return false if the path to create ends with a file seperator..even when the path has been successfully created. nice.
19 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 7781 [redacted]
craigc@cc_ev3_trunk edit (text+ko) 116917 - "wrq" -> "enterview"
20 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 8043 [redacted]
craigc@cc_ev3_trunk edit (text+ko) Fixed "loaded" debug message wiggy-ness
21 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 8079 [redacted]
craigc@cc_ev3_trunk edit (text+ko) uh, fixed a fix from earlier tonight..uh earlier last night.
22 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 8401 [redacted]
craigc@cc_ev3_trunk edit (text+ko) 117680 - made isVersionCompatible() compatible with design spec. Also added isDeveloperVersion(), which is used to special case developer builds in the cache.
23 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 8428 [redacted]
craigc@cc_ev3_trunk edit (text+ko) 117680 (part 2) - oops, forgot something.
24 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 8567 [redacted]
craigc@cc_ev3_trunk edit (text+ko) 117395 - My first hack at killing an ugiy JarLoader reentrancy problem. I synchronized a few critical operations: the construction of the JarLoader, and the construction of a class. So far, i haven't seen the bug since.
25 //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java 8694 [redacted]
craigc@cc_ev3_trunk edit; branch into //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java#1 (text+ko) 118079 - cant call createJarLoader() in NS. I HATE NETSCAPE. Fix (read: hack) is for createJarLoader() to return an Object, not a JarLoader. Boneheaded.
1 //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java 8865 [redacted] edm@edm-ws400_ev30_Beta
branch: branch into //EnterView/3.0_J42_Beta/Dev/com/wrq/jpm/JarLoader.java#1; branch from //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java#1,#25 (text+ko) Creating EnterView 3.0_Beta branch.

REDACTED

2 //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java 9025 [redacted] edm@edmws400_ev-all
integrate: copy from //EnterView/3.0_J42_Beta/Dev/com/wrq/jpm/JarLoader.java#2; merge into //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java#27 (text+ko) Integrating J42 Beta branch changes into EnterView 3.0 Beta branch.
3 //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java 9207 [redacted] edm@edm-ws400_ev30_Beta
integrate: copy from //EnterView/3.0_J42_Beta/Dev/com/wrq/jpm/JarLoader.java#3 (text+ko) Integrating J42 Beta changes into 3.0 branch.
4 //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java 9208 [redacted] edm@edmws400_ev-all
integrate: merge into //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java#28; merge from //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java#26,#27 (text+ko) Integrating Trunk into 3.0 branch.
5 //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java 9357 [redacted] maliaa@maliaa_410-EV30-beta edit (text+ko) 118412: Change security permissions under Netscape
6 //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java 9415 [redacted] craigc@cc_ev3_beta
edit; copy into //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java#29 (text+ko) 119698 - name change for local root directory 'enterview' -> 'reflectionweb'
7 //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java 9539 [redacted] craigc@cc_ev3_beta
edit (text+ko) 120289
8 //EnterView/3.0/Dev/com/wrq/jpm/JarLoader.java 9954 [redacted] craigc@cc_ev3_beta
edit; copy into //EnterView/Trunk/Dev/com/wrq/jpm/JarLoader.java#30 (text+ko) 120910

REDACTED

CLoader.java

```
package com.wrq.jpm;

import java.io.*;
import java.util.*;
import java.util.zip.*;
import java.net.*;
import java.awt.*;

public class CLoader extends ClassLoader implements IClassLoader {
  public final static String urlPrefix = "JPM";
  private static final String protocolPathProp = "java.protocol.handler.pkgs";
  private static boolean debug = false; // debugging
  private static boolean keepLoading = true; // aggresively load classes
  private String cookie;          // name of the jar file
  private ClassLoader parentLoader = null;

  /*
   * all instances, by cookie.  We really only instantiate one, see next, but
   * we are leaving the machinery around in case we need it again later
   */
  private static Hashtable loaders = new Hashtable();

  /*
   * The only CLoader we actually instantiate
   */
  public static CLoader ourLoader;

  /*
   * Additional directory from where to look for resources...
   * (after CLASSPATH and loaded JARs).
   * -- currently unused --
   */
  private String localResourceDirectory = null;

  /*
   * Overrides for local resources
   */
  private Hashtable localOverrides = new Hashtable();

  /**
   * Create a SipleClassLoader.  It is identified by a cookie string
   */
  [redacted]
  }

  /**
   * Resource and Mime Type HashTables
   */
  private Hashtable resourceHash = new Hashtable();
  private Hashtable mimeHash = new Hashtable();

  /** A hash table of AppletClassEntry's that define .class files in their
   * raw form.  We <B>do not</B> define these classes immediately
   * in defineClassFromBytes(), instead the byte arrays are kept
   * around and classes are defined lazily on demand, and the byte
   * array is removed from the Hashtable at that time.
   * <P>
   * By defining classes lazily, we prevent the case where a
   * derived class was found in a jar before its super class also
```

REDACTED

```
 * in the jar.  The derived class cannot be defined before it's
 * super class has even been loaded from the archive.
 *
 * classes are hashed by fully qualified name - "my.package.MyApplet"
 */

[redacted]


/**
 * Set some bytecodes as a definition for a class.
 * Do not actually define the class until later
 */

public void defineClassFromBytes(String name, byte[] buf) {
                rawClasses.put(name, buf);
}

/**
 * Define (& link) a class based on what was acquired previously in the JAR file
 * Should catch all exceptions and return non-null iff definition is valid
 */

[redacted]
// release the bytecodes...
[redacted]
                // We will print a message higher up, possibly before or earlier than
                // in this invocation pass through applyDefinition.
                return null;
            }
            // Check that the loaded class has the name we expect
            if (!c.getName().equals(name)) {
[redacted]
            }
            return c;
        }
}

/**
 * Define a class from a file
 */

p[redacted]
```

REDACTED

```
        InputStream is = new FileInputStream(fileName);
        int read = 0;
        while (read < length) {
          int r = is.read(buff, read, length-read);
          if (r < 0) {
                  break;
          }
          read = read + r;
        }
        return buff;
}

/**
 * Load a class from this class loader.
 *
 * @exception  ClassNotFoundException  if the class could not be found.
 */
public Class loadClass(String name) throws ClassNotFoundException {
        return loadClass(name, true);
}

/**
 * This is the main method for ClassLoaders, that is being redefined
 *
 * @exception  ClassNotFoundException  if the class could not be found.
 */

protected Class loadClass(String name, boolean resolve)
        t[redacted]

        /* We check to see if we've already loaded it (i.e. in the cache) */
        Class cl = findLoadedClass(name);

        //[redacted]

        // If the class hasn't already been loaded from the JAR,
        // we first try looking in the JAR:
        if (cl == null) {
          // First try to load the class from the JAR:
          [redacted]
        }

        [redacted])                                    \
        {
                [redacted];
                        [redacted]
                                [redacted]
                        else
                                [redacted]
                        return cl;
                }
                catch(ClassNotFoundException cnfe) {
                }
        }

        // If the class isn't in the JAR, try the system classloader:
        [redacted]
          try {
                [redacted]
                return cl;
          [redacted]
                // Drop through.
          }
        }
        if (cl == null) {
          t[redacted]
```

REDACTED

```
        }
        if (resolve) {
          resolveClass(cl);
        }
        [redacted]
        return cl;
    }

    /**
     * The resource stuff
     */

    /**
     * Assign an InputStream as the source for a given property name
     * This value comes first after the system resources
     */

    public void putClassResource(String name, String type) {
                    [redacted]
    }

    public void putLocalResource(String name, byte[] data, String type) {
        resourceHash.put(name, data);
        mimeHash.put(name, type);
    }

    public URL getResource(String name) {
        URL back = getSystemResource(name);
        if (back != null) {
          return back;
        }
        return getLocalResource(name);
    }

    public InputStream getResourceAsStream(String name) {
        InputStream back = getSystemResourceAsStream(name);
        if (back != null) {
          return back;
        }
        return getLocalResourceAsStream(name);
    }


    /**
     * Return a URL to the desired resource.
     */
    public URL getLocalResource(String name) {
        [redacted]
        if (o == null) {
          // Check on the JAR objects
          o = resourceHash.get(name);
        }
        if (o == null && localResourceDirectory != null) {
          // Try in localResourceDirectory
          File f = new File(localResourceDirectory, name);
          if (f.exists()) {
                    [redacted]
          }
        }
        if (o != null) {
          // Create a URL to refer to this resource
          try {
                URL url = new URL("simpleresource",
```

REDACTED

```
                                        "",
                                        "/"+urlPrefix+
                                        cookie+"/+/"+name);
                return url;
            } catch (Exception e) {
                debug("Exception "+e+" while building a resource URL");
                return null;
            }
        }
        return null;
    }

    private InputStream getLocalResourceAsStream(String name) {
        Object o;

        // Check data loaded from JAR
        [redacted]

        if (o != null) {
            [redacted]
                // This is a .class entry...
                // no access to .class files through getResource() in 1.1
                [redacted]
            }

            byte[] buf = (byte []) o;
            return new ByteArrayInputStream(buf);
        }
        if (localResourceDirectory != null) {
            // Now try in localResourceDirectory
            File f = new File(localResourceDirectory, name);
            try {
                return new FileInputStream(f);
            } catch (Exception ex) {
                return null;
            }
        }
        return null;
    }

    /**
     * Returns an InputStream on the resource
     */

    public static CLoader createLoader(String cookie, ClassLoader parent) {
                [redacted]

                if (back != null) {
                    if (back.parentLoader != parent)
                        th[redacted]
                    return back;
                }
                else {
                    [redacted]
                }
    }

/*
    pu[redacted]
                                                    String dir) {
        CLoader back = getLoader(cookie);
        [redacted]
            if (!back.localResourceDirectory.equals(dir)) {
                th[redacted]
            }
            return back;
```

REDACTED

```
        } else {
          [redacted]
        }
    }
*/
    private static CLoader getLoader(String cookie) {
        [redacted]
    }

    // get the local resource object...

    public static Object getLocalResource(String cookie,
                                          String name) {
        [redacted]

        // Check data loaded from JAR
        [redacted]
        String type = (String) cl.mimeHash.get(name);
        [redacted]
          // Came from JAR
          [redacted]

          if [redacted]
                // This is a .class entry...
                // no access to .class files through getResource() in 1.1
                [redacted]
          }

          byte[] buf = (byte []) o;
          [redacted]
                return ((Toolkit.getDefaultToolkit()).createImage(buf)).getSource();
          } else {
                [redacted]
          }
        }
        [redacted]
          // Check into localResourceDirectory
          [redacted]
          if [redacted]
                try {
                  URL url = new URL("file",
                                    "",
                                    [redacted]
                  return url.getContent();
                  [redacted]
                  [redacted]
                }
          }
        }
        return null;
    }

    // REMIND - si[redacted]

    public static InputStream getLocalResourceAsStream(String cookie,
                                                       String name) {
        CLoader cl = getLoader(cookie);
        [redacted]
    }

    /**
     [redacted]
     */
    private static void debug(String msg) {
        if [redacted]
          System.err.println("CLoader:: "+msg);
```

REDACTED

```
        }
    }

    static public ClassLoader makeClassLoader(String name, ClassLoader parent)
    {
        C[redacted]
        switch(Secure.getBrowser()) {
        [redacted]
c[redacted]
                [redacted]
                break;
            case Secure.NETSCAPE:
                [redacted]
                cl = CLoader.createLoader(name, parent);
                break;
            default:
                [redacted]
                break;
        }
        return cl;
    }

    /**
    [redacted]
    */
    [redacted]
        // Add this protocol type to the http properties
        try {
            P[redacted]
            newP.put(protocolPathProp, newP.getProperty(protocolPathProp)+"|com.wrq.jpm");
            S[redacted]
        }
        catch(Exception e) {
            [redacted]
        }

    }
}
```

JarCache.java

```
package com.wrq.jpm;

import java.io.*;
import java.util.*;

class JarCache extends Hashtable
{
  p[redacted]

        private File cacheDir = null;

        [redacted]
        {
                buildCache();
        }

        public InputStream getInputStream(String jarName)
        {
                [redacted]
                try {
                        if (ce != null)
                                r[redacted]
                } catch(Exception e) {}

                r[redacted]ll;
        }

        public CacheEntry find(String jarName)
        {
                [redacted]
        }

        public CacheEntry update(String jarName, InputStream jis) throws IOException
        {
                r[redacted]
                return persist(jarName, jis);
        }

        [redacted]
        {
                if (find(jarName) != null)
                        [redacted]

                CacheEntry ce = new CacheEntry(getCacheDir(), jarName, "000000000");
                [redacted]
                if (!(jis instanceof BufferedInputStream))
                        [redacted]
                byte buffer[] = new byte[1024];
                [redacted]
                for (;;) {
                        [redacted]
                        if (len <= 0) { break; }
                        fo[redacted]
                }
                fos.close();
                [redacted]
                return ce;
        }

        pu[redacted]IOException
        {
                debug("building cache");
                [redacted]
```

REDACTED

REDACTED

```
try {
        // Find the cache, or create it if necessary, and make sure we
        // have the appropriate access rights.
        //
        [redacted]
        [redacted]
        if (!Secure.exists(dir)) {
                [redacted]) {
                        debug("failed to make cache directory");
                        [redacted]
                }
        }
        else {
                [redacted]
                if (files != null) {
                        [redacted]
                        String jarVersion = null;
                        [redacted]
                        for(int i=0; i<files.length; i++) {
                                [redacted]

                                // Convert file name extension into more appropriate version string.
                                //
                                try {
                                [redacted]
                                jarVersion = file.substring(afterDot);
                                [redacted]
                                        CacheEntry e = new CacheEntry(dir, jarName,
jarVersion);
                                        [redacted];
                                        if (old == null) {
                                        }
                                        else {
                                                [redacted]
                                        }
                                } catch(Exception e) {
                                        [redacted]
                                }
                        }
                }
        }

        return true;
}
[redacted]
        debug("buildCache: " + e);
        [redacted]
        return false;
}
}

[redacted]
{
        CacheEntry e = (CacheEntry)super.remove(jarName);
        [redacted] {
                Secure.fileDelete(e);
        }
        [redacted]
}

private File getCacheDir()
{
        [redacted]
                try {
                        // Ask for privileges here!
                        [redacted]
```

```
                    if (buf.length() != 0)
                    {
                                [redacted]
                                        buf.append(File.separatorChar);
                                [redacted]
                                buf.append(File.separatorChar);
                                [redacted]
                    }
                [redacted]
            }
            return cacheDir;
        }

    /**
     [redacted]
     */
    private static void debug(String msg) {
                [redacted]
                    System.err.println("JarLoader:: "+msg);
    }
}
```

REDACTED